sensors

MDPI

Article

Five-Direction Occlusion Filling with Five Layer Parallel Two-Stage Pipeline for Stereo Matching with Sub-Pixel Disparity Map Estimation

Yunhao Ma †, Xiwei Fang †, Xinyu Guan, Ke Li, Lei Chen * and Fengwei An *

School of Microelectronics, Southern University of Science and Technology, Shenzhen 518055, China
* Correspondence: chenl33@sustech.edu.cn (L.C.); anfw@sustech.edu.cn (F.A.)
† These authors contributed equally to this work.

Abstract: Binocular stereoscopic matching is an essential method in computer vision, imitating human binocular technology to obtain distance information. Among plentiful stereo matching algorithms, Semi-Global Matching (SGM) is recognized as one of the most popular vision algorithms due to its relatively low power consumption and high accuracy, resulting in many excellent SGM-based hardware accelerators. However, vision algorithms, including SGM, are still somewhat inaccurate in actual long-range applications. Therefore, this paper proposes a disparity improvement strategy based on subpixel interpolation and disparity optimization post-processing using an area optimization strategy, hardware-friendly divider, split look-up table, and the clock alignment multi-directional disparity occlusion filling, and depth acquisition based on floating-point operations. The hardware architecture based on optimization algorithms is on the Stratix-IV platform. It consumes about 5.6 K LUTs, 12.8 K registers, and 2.5 M bits of on-chip memory. Meanwhile, the non-occlusion error rate of only 4.61% is about 1% better than the state-of-the-art works in the KITTI2015 dataset. The maximum working frequency can reach up to 98.28 MHz for the 640 × 480 resolution video and 128 disparity range with the power dissipation of 1.459 W and 320 frames per second processing speed.

Keywords: semi-global matching; subpixel interpolation; multi-direction occlusion filling; single precision floating point; disparity refinement; FPGA

Citation: Ma, Y.; Fang, X.; Guan, X.; Li, K.; Chen, L.; An, F. Five-Direction Occlusion Filling with Five Layer Parallel Two-Stage Pipeline for Stereo Matching with Sub-Pixel Disparity Map Estimation. *Sensors* **2022**, *22*, 8605. https://doi.org/10.3390/s22228605

Academic Editor: Radu Danescu

Received: 29 August 2022
Accepted: 4 November 2022
Published: 8 November 2022

1. Introduction

Binocular stereoscopic matching is an essential method in computer vision, [1,2] imitating human binocular technology to obtain distance information. The main goal of binocular stereoscopic matching is to find the corresponding point from two images of the same scene and use the similar triangle principle to generate a reference image disparity map. Based on the disparity diagram generated by binocular stereo matching, a depth map for 3D reconstruction can be generated based on spatial geometric relationships [3,4]. The binocular stereo vision technology has been applied to many fields, including medical diagnosis, unmanned driving [5], virtual reality, three-dimensional reconstruction, robot navigation [6], drone piloting, and virtual reality and augmented reality (VR/AR) applications in need of disparity estimation [7].

1.1. Related Works

Among plentiful stereo matching algorithms, Semi-Global Matching (SGM) is recognized as one of the most popular vision algorithms due to its relatively low power consumption and high accuracy, resulting in many excellent SGM-based hardware accelerators. H. Hirschmuller [8] presented stereo processing using SGM and mutual information on accurate applications. P. Dong et al. [9] proposed a coprocessor with a pixel-level pipeline and region-optimized method for semi-global matching in real cases and complimented it on FPGA.

However, vision algorithms, including SGM, are still relatively inaccurate in actual long-range applications. Compared with the algorithm based on deep learning, the traditional binocular stereo-matching algorithm is too inaccurate. Still, it cannot cope with high-complexity scenarios, so its application has been minimal. Specifically, under the condition of occlusion and mismatch, the algorithm generally has an undesirable situation with an increased error rate. These two points are crucial for practical application.

In recent years, stereo matching has still been paid massive attention to instead of disparity refinement, which is significant in obtaining precise depth information. Massive relative works focus on subpixel interpolation. Rui Fan et al. [10] proposed a disparity map with subpixel resolution where a disparity error larger than one pixel may result in a non-neglected difference in the reconstructed road surface.

Besides, Typical disparity filling considers only single row 2-direction filling: the 0-angle and 180-angle filling. Z. Chen et al. [11] offered a post-processing structure of occlusion filling in only 0-angle and 180-angle directions with an error rate of 7.27% using the same database and disparity range. S. Jin et al. [12] presented a left-right check and occlusion filling with a maximum working frequency 93.09 MHz and 11,000 slices, an exceeded resource utilization. Cambium et al. [13] got a disparity whose error rate is 32.9% under KITTI 2015 dataset after occlusion filling with 24.5 K LUTs and 9.1 K registers.

1.2. Contributions

This work proposes a disparity improvement strategy with at least 2× smaller FPGA resource usage compared to the previous works. as for the subpixel interpolation, the contribution can be featured in the hardware-friendly divider, split look-up table for cosine function. Furthermore, the disparity optimization post-processing novelty includes the clock alignment multi-directional disparity occlusion filling and depth acquisition based on floating-point operations. Finally, a hardware-friendly architecture is implemented on the FPGA platform with outstanding accuracy and robustness. The hardware architecture based on optimization algorithms is on the Stratix-IV platform. It consumes about 5.6 K LUTs, 12.8 K registers, and 2.5 M bits of on-chip memory. Compared with the state-of-the-art works, the error rate in non-occlusion is reduced by 1% under KITTI 2015 dataset.

1.3. Paper Structure

The remains of this paper are organized as follows. We illustrate the proposed subpixel interpolation and disparity optimization post-processing in Section 2. Section 3 elaborates on the implementation of the hardware architecture. Section 4 presents the experimental results with accuracy, hardware-resource usage, and performance. Finally, we conclude in Section 5.

2. Algorithm

2.1. Framework Overview

The overall flow from cameras to disparity depth map rendering is shown in Figure 1. The main contribution of this work is the post-processing procedure after the aggregation cost (marked with a star) in Figure 1. First, after image corrections and aggregation cost, the proposed sub-pixel interpolation calculates the decimal parts with a cosine look-up table and Newton's division operator. Further, by carefully conducting a left-right check for the occlusion and mismatching regions, this work processes the left and right disparity values into a multi-direction occlusion filling module for practical disparity values obtained according to different filling rules for occlusion and mismatching regions.

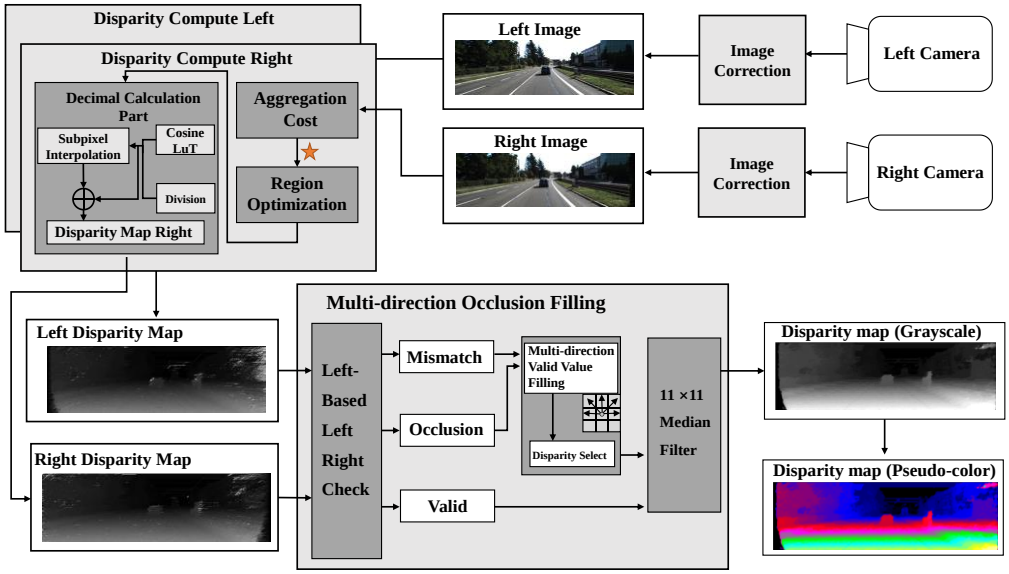

Figure 1. The overall framework of disparity refinement.

2.2. Aggregation Cost with Census Aggregation and Hamming Distance

Zabih et al. [14] proposed a commonly acknowledged census algorithm for calculating matching costs. It can significantly detect the local structural features and achieve high robustness in a light-variable environment. The including census transformation works well in both light and dark conditions. The aimed census flag has been obtained by comparing the gray value of each pixel in the adjacent window to the gray matter of each center pixel of the window. The corresponding explanation Equation containing census transform vector R is shown below:

$$R(P(i,j)) = \bigotimes_{(a,b)\epsilon W} \xi(P(i,j), P(a+i, b+j)) \\ \xi(p,\ p') = \begin{cases} 1,\ p < p' \\ 0,\ p \geq p' \end{cases} \tag{1}$$

where $\otimes$ denotes the concatenation, $P(i,j)$ means the center pixel, and W is the matching window. On the condition that the gray value of a certain pixel is greater than the gray value of the center pixel, it is marked as 1. (Otherwise, it is marked as 0.)

The Hamming distance shown in Figure 2 shows that the number of the corresponding bits of two-bit strings is not the same. The calculation method is to perform the XOR operation on the two vectors to obtain a new vector and calculate its number, as shown in Equation (2). The calculated one's number is the Hamming distance, which represents the initial matching cost of a pixel.

$$C\left(P_{(i,\,j)},\ d\right) = \sum Hamming(R_R\left(P_{(i-d,\,j)}\right), R_L(P_{(i,\,j)})) \tag{2}$$

In this case, $P_{(i,\,j)}$ means the position of a pixel in the base image, and $P_{(i-d,\,j)}$ means the position of the pixel in another image.

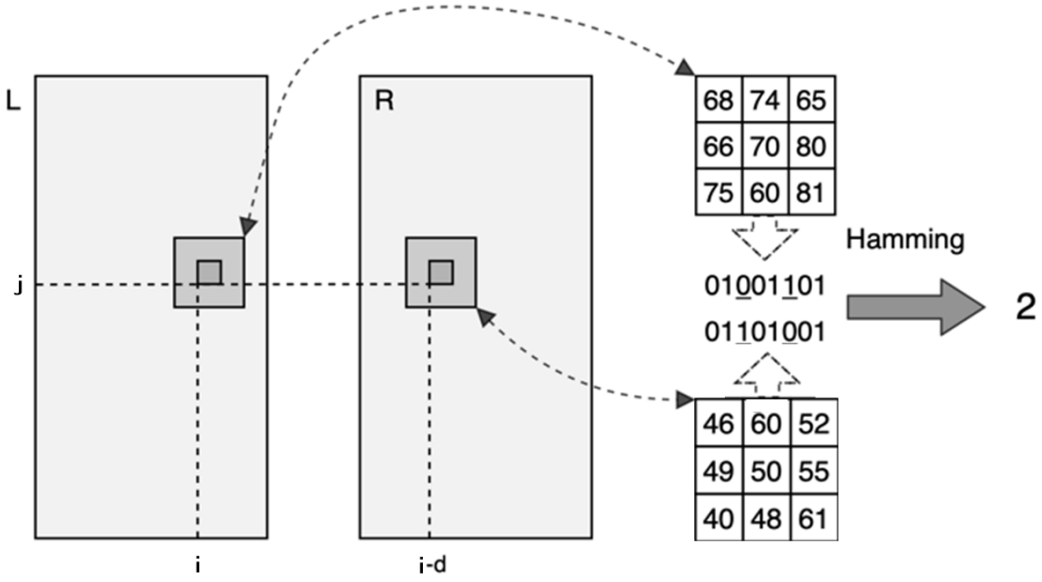

Figure 2. Schematic diagram of the initial matching cost calculation.

The Optimization step of the initial matching costs uses a cost aggregation strategy. In the SGM algorithm, a general energy function is established to minimize it for optimization. Further, it considers the one-way dynamic programming method to solve a two-dimensional optimization problem with the most optimized energy function. 0-angle and 135-angle directions resulted in the lowest error rate in experimental results from P. Dong [9], as shown in Figure 3.

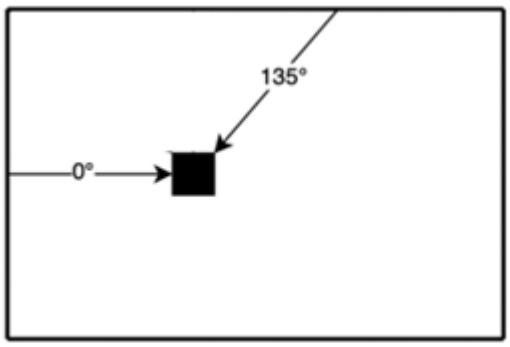

Figure 3. Cost aggregation path.

2.3. Disparity Computing with Subpixel Interpolation Based on Split Cosine Look-Up Table and Practical Divider

Semi-global matching uses the Winner-Take-All (WTA) method to calculate disparity [15]. Each pixel chooses the disparity value related to the minimum aggregate cost as the final disparity. This disparity is usually integer, not adequately reasonable, and desirable in actual cases. The integer is suitable for most circumstances but cannot consider the long-distance situation. According to Equation (7), even if the fraction part's value is relatively small, the depth can become significant with a low error rate after conducting the multiply. I. Haller and S. Nedevschi [16] proposed an approach of disparity interpolation to solve this problem. Interpolation can increase the decimal place of the disparity, which can effectively improve the accuracy at the line of sight. The difference is declared as the following Equation (3), which consists of two parts. Where d indicates the integer disparity, $interpFunction(i)$ indicates the interpolation value with essential fractions information, and m is the value obtained by the cost aggregation of the pixel.

$$d_{Final} = d + interpFunction(i) \tag{3}$$

$$leftDif = m_{d-1} - m_d \tag{4}$$

$$rightDif = m_{d+1} - m_d \tag{5}$$

$$i = \frac{rightDif}{leftDif} \tag{6}$$

$$interpFunction(x) = 0.5 - \frac{1}{2} \times cos\left(x \times \frac{\pi}{2}\right) \tag{7}$$

It is important to consider worst-case errors to make the disparity calculation method more robust. Therefore, this work uses the function with the maximum error instead of the sum of errors as the metric for function fitting in Equation (7).

This results in the demand for a hardware-friendly look-up table for the cosine function. Initially containing 360°, this look-up table is divided into six sub-tables, each 64° for better use. Here a projection of processes is formed; for instance, each y corresponds to a function value of a 9-decimal precision cos y, with each projected value effectively expressed through hardware description. Simultaneously, this part introduces the divider with Newton's method to simplify the calculation process. The representation of the denominator is crucial in most cases. However, it can be replaced as several local changing elements with small intervals typically ranging from one to two, where the parameter can be recorded as binary exponential forms. Finally, the interpolated disparity is obtained.

Further, this paper proposes a decimal replication method for the fraction part in interpolation. Figure 4 denotes the instances of the decimal replication method. For example, the result of the calculation may have many decimal places. To reduce the consumption of hardware resources, only two decimal places keep retained after the calculated decimal point, and the remaining decimal places are filled by copying the existing decimal places.

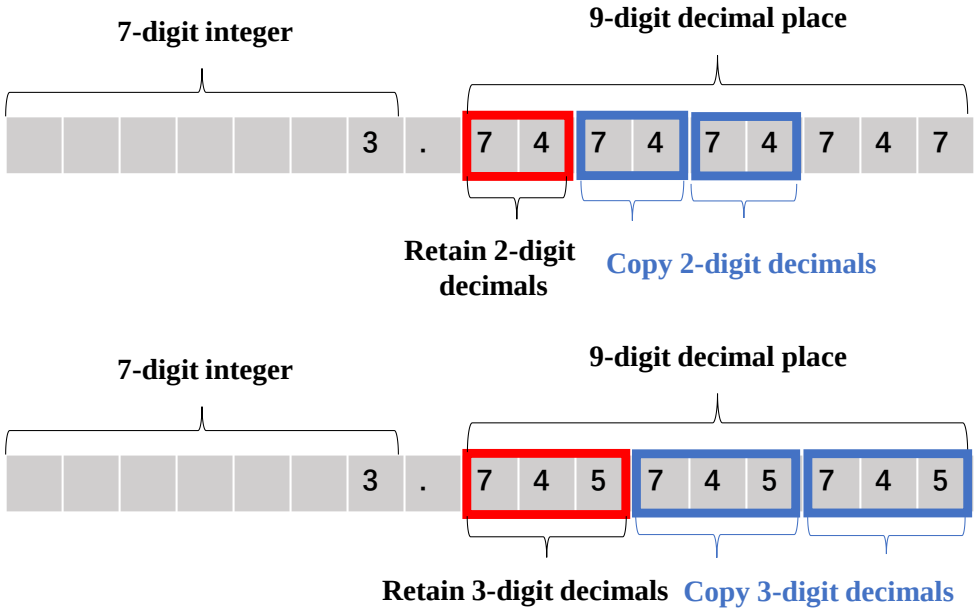

Figure 4. Examples of decimal replication method.

In Figure 5, the accuracy in retaining only 3-digit decimals is smaller, and the accuracy error with 3-digit decimals or more is almost the same. The accuracy improvement caused by more than three decimal places is tiny but requires more hardware resources. Therefore, 3-digit decimals and replicating them to other decimal places can balance the best accuracy and the minor resource consumption simultaneously.

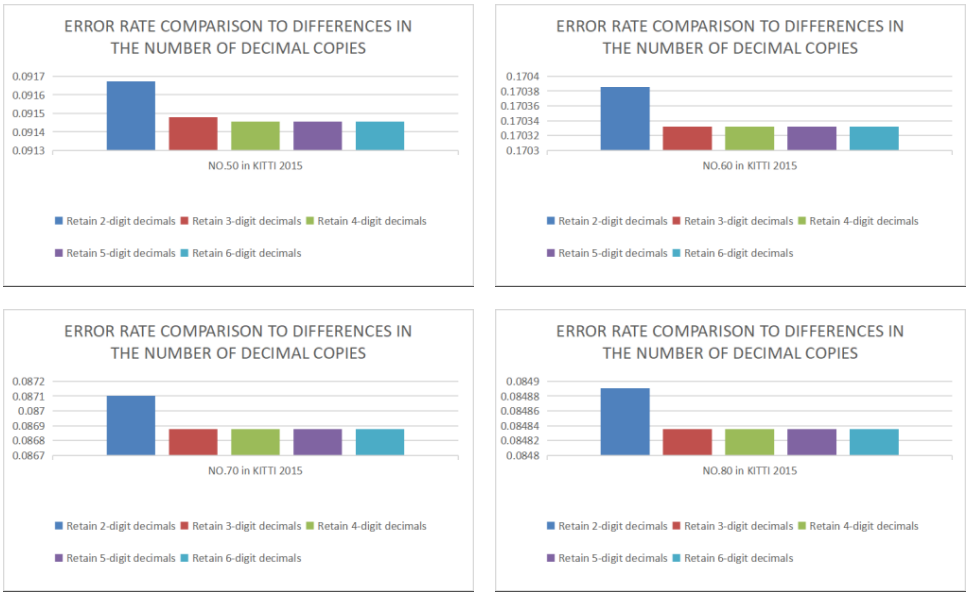

Figure 5. Verification of copied decimal parts.

2.4. Multi-Direction Disparity Occlusion Filling with Clock Alignment after Left-Right Check

Disparity filling is the main challenge in disparity refinement due to weak texture conditions, obstacles, etc. The mechanism of left-right detection is as follows: in calculating disparity, the disparity values of left and right maps at the same position are completely different. However, not all differences are unreasonable, and the calculated disparity is reasonable under the threshold. If exceeding, the disparity result is not reasonable. Disparity $D_L(i,j)$ stands for the disparity value of the pixel (i,j) in the left disparity map whereas $D_R(i,j)$ stands for the value of pixel (i,j) in the right one.

According to the left image and disparity uniqueness, $D_R(i,j)$ can be obtained by swapping the positions of the left and right images and then gaining the pixel with the same name as an individual pixel in the right image and the disparity value corresponding to the pixel. If the difference between these two disparity values is smaller than one pixel, it can be held. Otherwise, it is abandoned, and [17] proposed to divide it into mismatch or occlusion. The occlusion region is an area of pixels visible on the left view but not on the right view due to foreground occlusion. The occlusion region is more likely in the disparity discontinuity area, where one side is the foreground (larger disparity value) while the other is the background (smaller disparity value). The framework above can be shown in the following Equation (8).

$$|D_L(i,j) - D_R(i - D_L(i,j),j)| \leq 1 \quad (8)$$

Generally, the disparity of the surrounding background pixels should be chosen for occlusion conditions when filling occlusions, and the foreground pixels should be avoided [8]. Because the background pixel disparity value is smaller than the foreground, the adjacent minimum is chosen by collecting surrounding valid values. The contiguous pixels of the mismatching region are most likely on the consecutive surface. All pixels in the adjacent area are expected to be concerned when processing occlusion filling. Since the valid disparity values around the mismatching region are relatively equivalent, the middle one is optional. Equation (9) expresses how a resolution-dependent size median filter works for hole-filling. Each hole chooses five valid pixels and then sorts these values from largest to smallest, as shown in Figure 6. If a hole is an occlusion, the second smallest of the five pixels is selected to fill it; if this hole is mismatched, the median of the five pixels is chosen to fill it.

$$D\prime_p = \begin{cases} second\ minimum\ v_{pi}\ \ occlusion \\ median\ v_{pi}\ \ mismatch \\ D_p\ \ other\ cases \end{cases} \tag{9}$$

Here, the v_{pi} means the sequence of five-direction valid disparity from minimum to maximum and D_p is a defined value set in the beginning.

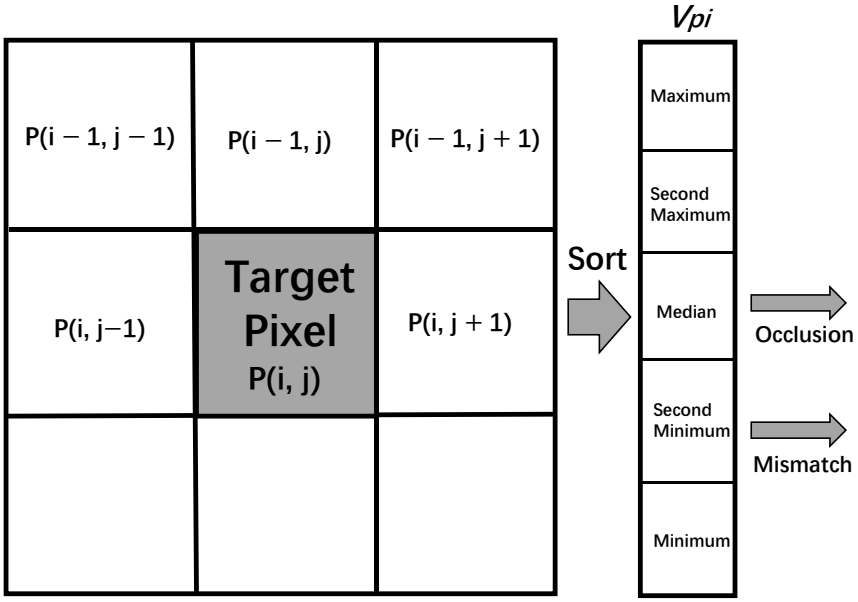

Figure 6. Valid values for occlusion and mismatch.

2.5. Floating-Point Operation for Disparity Conversion Depth

Binocular stereo vision fuses the images obtained by the two eyes and observes the differences between them so that we can obtain a distinct sense of depth, establish a correspondence between features, and correspond to the reflection points of the same spatial, physical point in different images. Such a difference, disparity image is defined.

Depth image, also known as distance imagery, refers to an image that takes the distance (depth) value from the image collector to each point in the scene as the pixel value. Thus, to better clarify the depth information of the camera input data, adopting the disparity conversion depth function, Equation (10) is shown below:

$$depth(P(i,j)) = \frac{fB}{cZ} \tag{10}$$

where depth($P(i, j)$) is the depth of $P(i, j)$, c is the pixel size, B is the baseline, f is the pre-calibrated focus length, and Z is the depth of the point from the camera. In this case, we get a more intuitive depth of information instead of a disparity value.

3. Hardware Implementation

The hardware architecture in Figure 7, emphasizing post-processing disparity optimization, consists of the initial aggregation costs and post-processing. The refined disparity map and the corresponding depth map are expressed in single-precision floating point numbers.

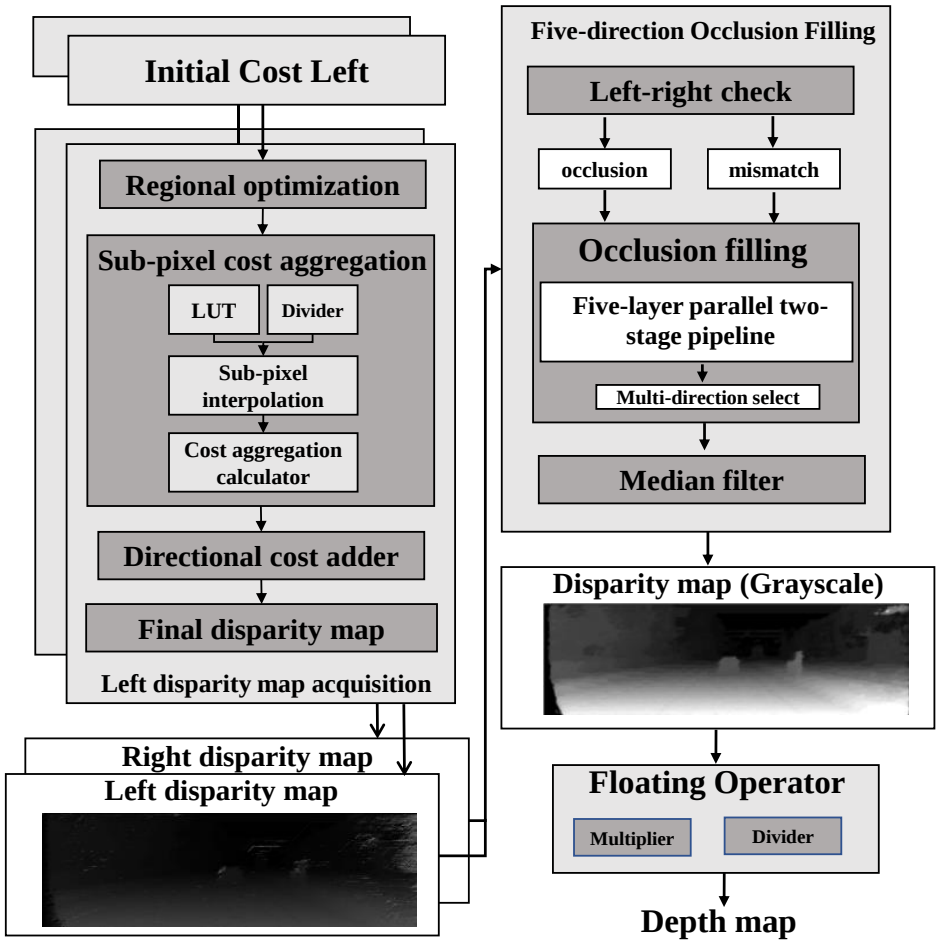

Figure 7. The hardware implementation for sub-pixel cost aggregation and five-direction occlusion filling.

3.1. Sub-Pixel Interpolation

Figure 8 demonstrates the hardware architecture of the sub-pixel interpolation module. (1): enter 32 aggregate values in the two directions, 0°, and 135°. To be able to port on the hardware and ensure accuracy, this work innovatively proposed a processing idea of regional optimization. The 128 initial (usual cases) costs are treated in groups of four, and each group, the smallest generation value is considered the overall generation value of the group. Take Figure 9 as an example. The 12-generation values are divided into three groups. The minimum generation values of each group are found at 94, 102, and 107, and the relative positions of these three generation values in each group are 2, 3, and 0, respectively, so the minimum generation value is spliced with the relative position to obtain three regional optimization values {94, 2}, {102, 3} and {107, 0}, through which the algorithm resource consumption can be greatly reduced. Its advantages are reflected in the process of resource utilization and subsequent cost aggregation.

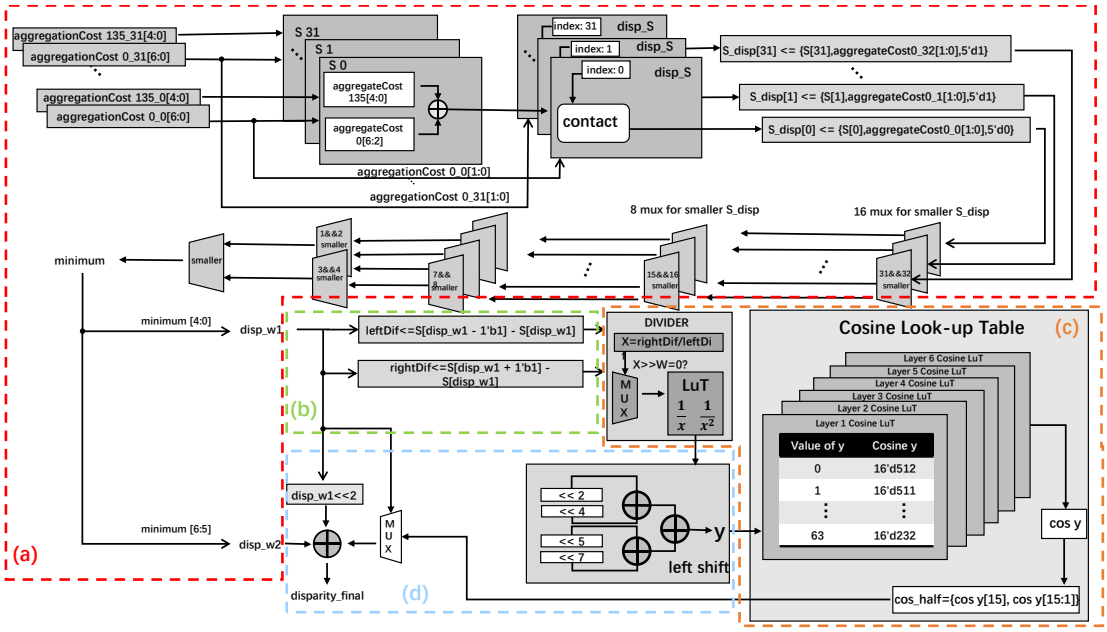

Figure 8. Subpixel interpolation for disparity computation.

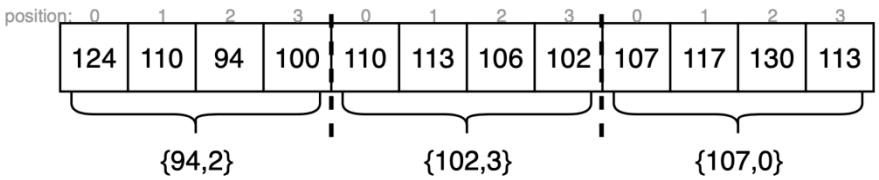

Figure 9. Example of region optimization.

In Figure 8a, adding aggregated values in all directions can accumulate the matching costs for every pixel. The smallest regional generation value is found through 5 sets of selectors. This minimum region of the minimum optimized cost output is disp_w1, and the position flag output is disp_w2.

As shown in (b) of Figure 8, the difference between the value of the smallest region and the value of the smallest adjacent region is calculated, and the two calculated differences are recorded as leftDif and rightDif, respectively.

In (c) of Figure 8, the result, y, is obtained after division and moving the quotient through the shift operation. Subsequently, using the cosine look-up-table, find out the cosine value corresponding to y, and the result COS_half is obtained by shifting the cosine value right by 1 bit.

Finally, in Figure 8d, the disp_w1 is used to determine whether the region is on edge. If it is not an edge region, multiply the region with the minimum optimized cost by the number of costs in a specific region and add it to the tag flag and the interpolation Equation of intern fuction = 0.5 − COS_half to get the final subpixel disparity value.

3.2. Left-Right Check

Figure 10 demonstrates the hardware architecture of the left-right check module. The left-right check module ensures the matching pixels' effectiveness in accomplishing the filling operation. Based on the inputs and the settlement of the disparity range, 2 SRAMs whose depth is 256 bits are controlled for bit-level reading and writing. The current address of the left disparity and the value are needed to determine the correct address index for the right SRAM to search.

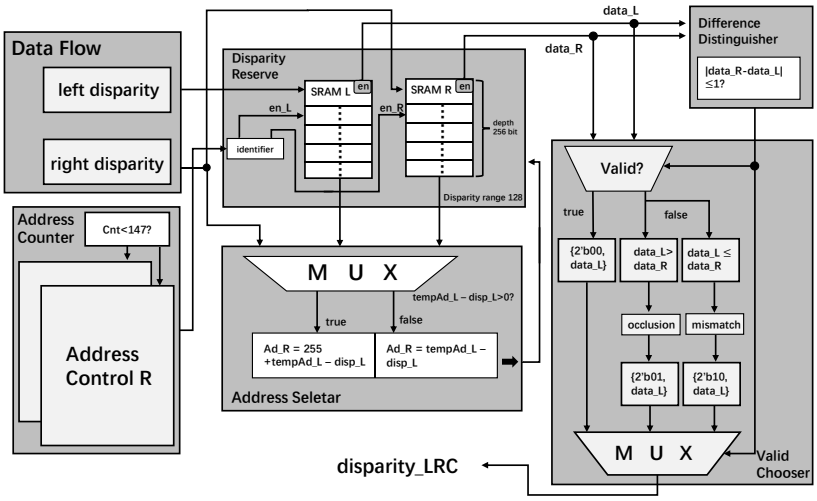

Figure 10. The Left-right Check Architecture.

We find the disparity valid whether the absolute difference between left and right disparities is within the threshold range and obtain the output disparity_LRC from the combination of flag bits 2'b00 and the original left disparity. Reversely, on the condition that the difference where the left disparity is numerically bigger than the right one (occlusion) is beyond the threshold, the disparity is invalid, and disparity_LRC is set as a combination of flags bits 2'b10, and the original left disparity. Otherwise, we meet the mismatch condition where the disparity_LRC should be 2'b01 with the original left disparity.

3.3. Alignment Multi-Direction Occlusion Filling

Initially, if the output of the previous module's left-right check is valid already, the output disparity_filling is set at the same value as the input value. Otherwise, the final disparity value should be gained through adjacent disparity values around the current pixel under different cases. Here we name the process alignment multi-direction occlusion filling described in Figure 11. The first two bits of inputs are cashed to determine different conditions.

(1) On the condition that the input flag is invalid, the data flow first enters both a LIFO and a FIFO, which hold the same resolution aligned all the time. Then, these pixel coordinates begin to find 180° disparity, 45° disparity, and 90° disparity simultaneously. The value of 180° can be easily gotten through the next pixel coordinate on the condition that the disparity value here is valid. However, the fetch of the 45° and 90° values is relatively tricky. Figure 12 shows a 45° disparity in the northwest direction $P(i-1, j-1)$, while the 90° disparity is gained through the absolute above value $P(i-1, j)$. In the hardware framework, 90° disparity is achieved in the southern direction because of the same length of the FIFO, where the previous pixel will go further in this hardware architecture. Similarly, a 45° disparity is obtained from the southeast direction through this principle.

(2) In the next clock cycle (the second yellow dashed line in Figure 8), on the one hand, the 180° valid disparity value is cashed while the original reversed data flow is retained as well. On the other hand, 135° disparity is obtained through the southwest direction hierarchy.

(3) Disparity values enter another group of FIFO and LIFO, which is used for new line buffers and data initiation, respectively. In hierarchy 3, all registers sustain their values, while a 0° disparity valid value is also found according to adjacent pixels.

(4) Due to the high-standard alignment of the data stream, in the next cycle, five valid disparity values from five directions ranging from 0° to 180° are attained in the select value module. A high parallelism combinatorial logic bubble sort is designed to sort the disparity values in five directions from small to large. Firstly, two adjacent indexes

except five values are compared two by two with a result sequence from small to large. Second, two adjacent indexes except index 1 of values are also compared with the result sequence from small to large. After five loops of such logic combination, the final sequenced output disparity value appears, where the median value is selected for occlusion while the second minimum value is selected for mismatch as the final disparity_filling output.

(5) Finally, the output is filtered with a resolution-dependent median filter for better margin information.

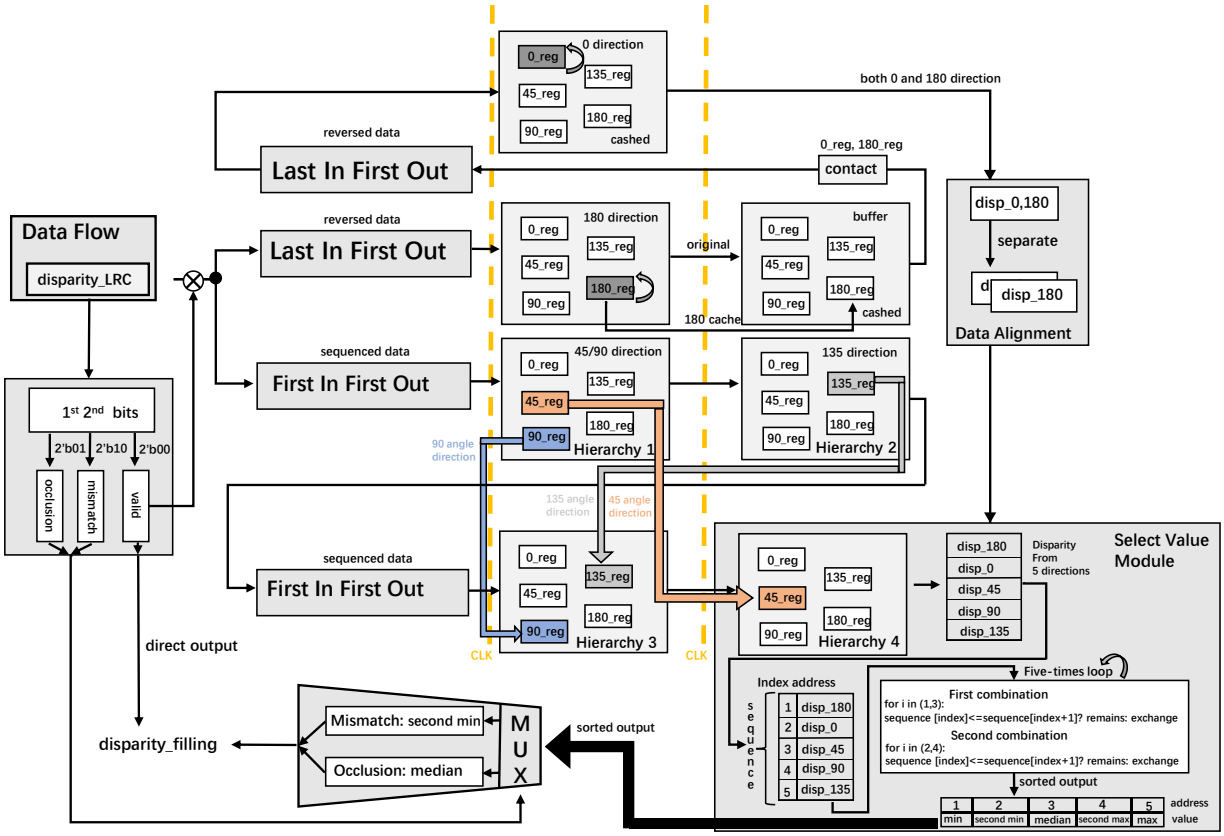

Figure 11. The alignment multi-direction occlusion filling module.

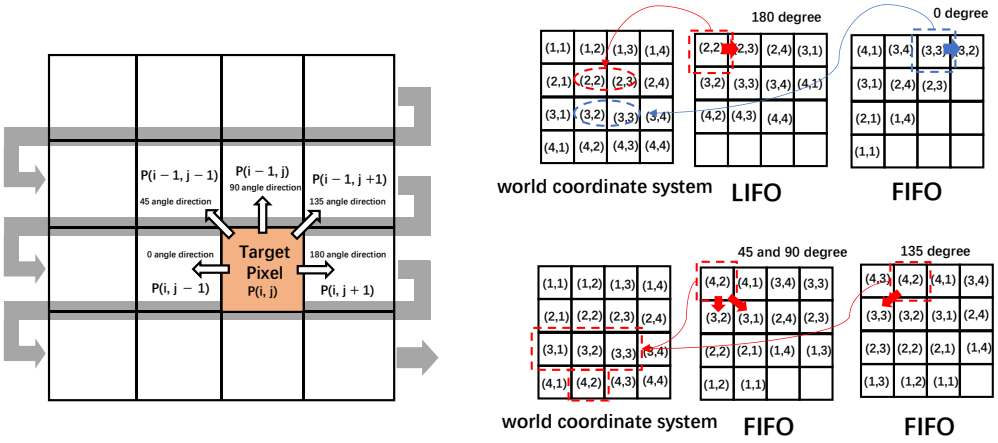

Figure 12. The direct sketch map of multi-direction occlusion filling.

3.4. Floating Point Operation Process

In this work, only a few meaningful bits of disparity are retained. The other extra bits of disparity are filled with these meaningful bits repeatedly. In this case, there needs only a small number of resources while confirm accuracy.

Figure 13 demonstrates the architecture of the floating-point operation process. When performing multiplication and division of single-precision floating point numbers, we propose a special floating point number pipeline multiplier and division device. Based on the IEEE 754 format, the floating-point number of the 32-bit consists of a 1-bit symbol bit (S), an 8-bit exponent bit (E), and a 23-bit mantissa bit (M). When multiplying the floating-point numbers, the exponent bits of the multipliers are added first and multiplied by the mantissa bit (1.M) of the same number of bits. When performing the division operation, the same as the multiplication, the divisor's exponent is first minus the exponential bit of the divisor, and the decimal number (1.M) of the dividend is divided by the decimal number (1.M) of the divisor. The result float number is represented in the single-precision format after standardizing.

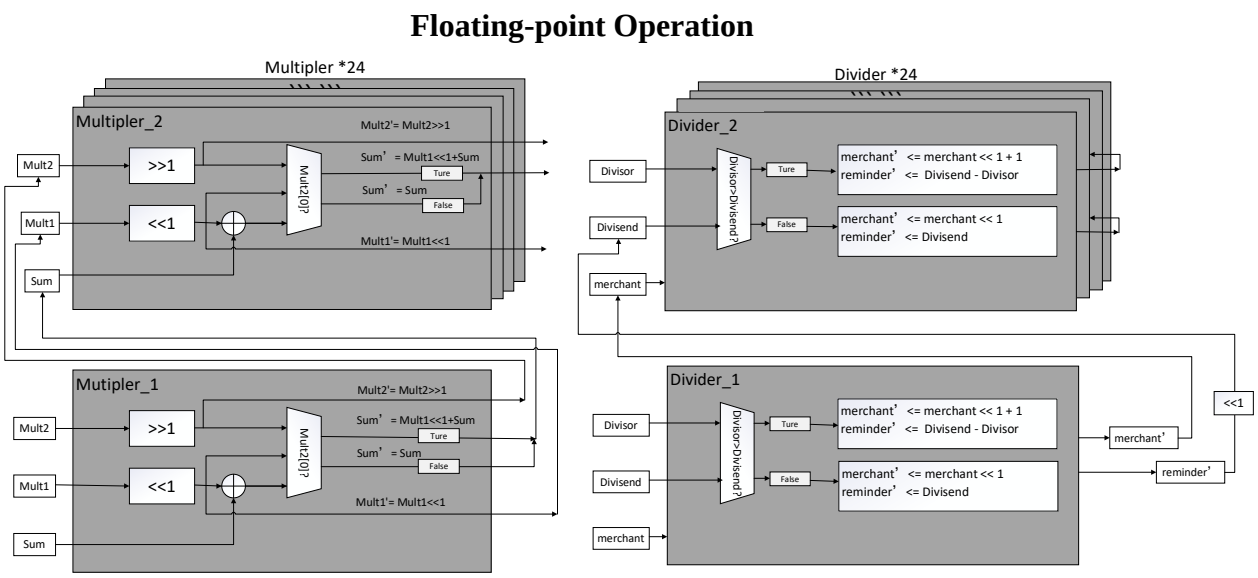

Figure 13. The floating-point operation consists of division and multiplier.

3.4.1. Floating-Point Multiplier

In the design of this paper, the pipeline consists of several cells. A cell is responsible for an addition operation, which outputs the cumulative sum with the shifted multiplier and serve as the next cell's input. The input for each cell is multiplier 1 and multiplier 2. First, shift multiplier 1 to the left by one bit and multiplier 2 to the right by one bit. If the lowest bit of multiplier 2 is 1, the left-shifted multiplier one is added to the previous cumulative sum; if the lowest bit of multiplier 2 is 0, the cumulative sum does not change. After the execution in this cell, the shifted multiplier 1, 2, and the new cumulative sum are output.

The first cell is first initialized with the original multiplier 1, 2. Since multipliers 1 and 2 have 24-bit, 24 addition operations and 24 cell modules are generated. The product of the outputs is a fixed 48-bit in general, such as $10.M_3$, $01.M_3$, $11.M_3$.

Under the IEEE 754 standard, the leading number of the significand is always 1. Consequently, a leading one can be implied, and the explicitly represented part of the significand lies between 0 and 1. After normalization and rounding to the nearest value, the 48-bit product represents in the standard format, e.g., 24-bit 1.M.

3.4.2. Floating Point Divider

Similarly to the multiplier above, a cell is responsible for a single subtraction operation and shift operation. If the divisor is less than or equal to the dividend, the dividend is minus the divisor, and the quotient shifts left one place and added 1 to the lowest bit. If the divisor exceeds the dividend, the quotient shifts left one place, and the dividend becomes the remainder. The remainder of the last cell outputs and shifts left one place, then serves as the input dividend of the next cell.

The first cell is initialized with the original divisor and dividend. Since both the divisor and the divisor are 24-bit, 24 subtraction operations will be performed, and 24 cell modules will be generated. The final quotient is fixed 24-bit in general, either $1.M_3$ or $0.M_3$. Furthermore, the quotient will be represented in the standard IEEE 754 format, e.g., 1.M, like the product's rounding and normalization.

4. Results and Discussion

The proposed algorithm is implemented on Intel Stratix IV (EP4SGX230) and Stratix V (5SGXEA7N2F45C2N) FPGA in Verilog HDL. We leverage the simulation results on the KITTI 2015 data sets to validate the accuracy. The hardware resource usage and power dissipation can be found in the synthesis results from the FPGA EDA tool, namely Quartus Prime. In this work, the disparity range is set to 128 because it is a typical value in the previous works for both VGA and XGA video input. Besides, we take VGA resolution for the fare comparison with the convincing hardware resources in [12,13].

The simulation results in Figure 14 show the gray disparity map, pseudo-color disparity map, and error map of NO.110 and NO.128 images in KITTI 2015. Figure 15 shows the proposed gray disparity map of NO.174 image in KITTI 2015 compared with only 0 and 180-angle directions filling with the original disparity map without post-processing. From error rates of 7.27% in two-direction single-row filling and 6.04% in presented five-direction spanning multiple rows, the filling efficacy increases, especially in the red box. The quality of the disparity map is enhanced through five-direction spanning multiple rows, where the boundaries are more refined and less affected by the surroundings.

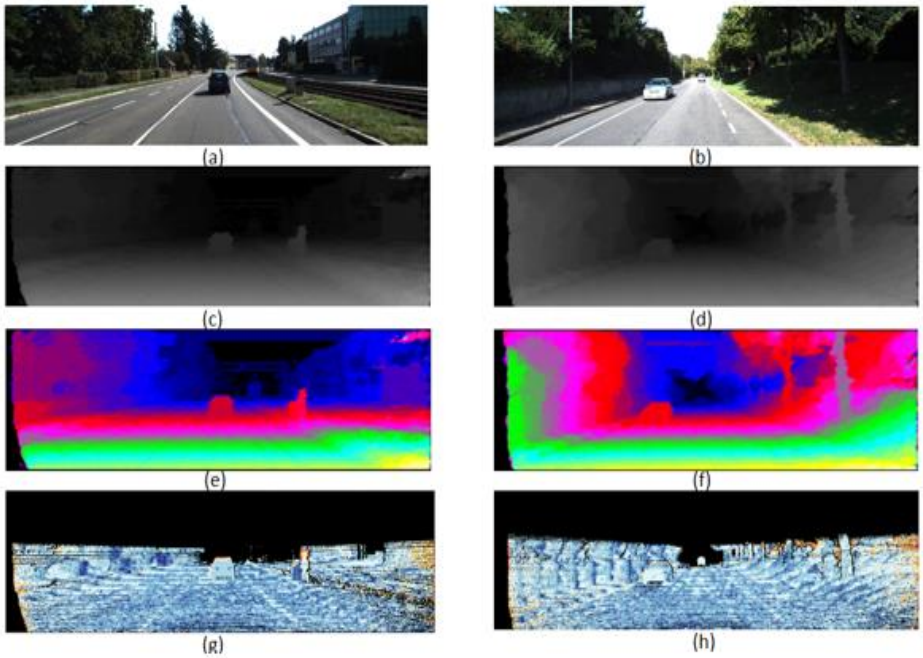

Figure 14. (**a**) No.110; (**b**)No.128; (**c**) Grayscale disparity of (**a**); (**d**) Grayscale disparity of (**b**); (**e**) Pseudo-color disparity of (**a**); (**f**) Pseudo-color disparity of (**b**); (**g**) Error of (**a**) (noc:2.77%, occ:4.48%); (**h**) Error of (**b**) (noc:3.80%, occ:6.06%).

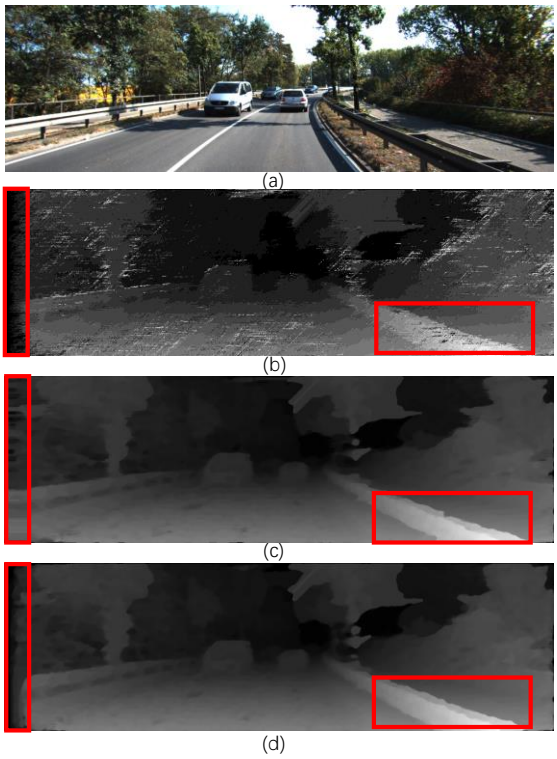

Figure 15. Disparity map comparison of (**a**) original No.174 image in KITTI 2015; (**b**)The disparity map without post-processing; (**c**) use only 0 and 180 angle direction filling with error rate 7.27%; (**d**) use proposed five-direction filling with error rate 6.04%.

In Table 1, considering this work aims at disparity refinement, we only compare the hardware resources of the post-processing module in previous works since this part of the contribution is relatively scarce in the current study.

Table 1. Comparison of FPGA resources (-: not mentioned).

	This Work		**[12]**	**[13]**
Resolution	**640 × 480**	640 × 480	640 × 480	640 × 480
Disparity Range	**128**	128	64	128
FPGA Platform	**Stratix-IV**	Stratix-V	Xilinx XST J.33.	Stratix-IV
LUTs	**5.6 K**	5.76 K	60 K	12.6 K
Registers	**12.8 K**	12.9 K	-	9.1 K
On-Chip Memories (bits)	**2.5 M**	2.5 M	2.06 M	2.8 M
Frame per Second (fps)	**320**	375	302	-
Frequency (MHz)	**98.28**	115.3	93	-
Power Dissipation (W)	**1.459**	0.876	-	-

In Table 2, these compared works propose an entire system where the inputs are camera streams and outputs are disparity videos. Unlike them, this work contributes to disparity refinement; therefore, we only compare the output error rate of the proposed work with plentiful advanced works from journals and conferences under non-occlusion and occlusion situations. We take the 20 best error rates of images from the KITTI 2015 dataset and obtain their average. Generally, this work has achieved an accuracy of 4.61% in the non-occlusion situation and 6.04% in the occlusion situation. The presented work offers more precise results than the experimental measurements in [9,18,19] for non-occlusion conditions, whose error rates are 6.54%, 5.66%, and 6.58%, respectively. Meanwhile, for occlusion consideration, our work has a lower error rate than the error rate of 7.52% from [20], 6.88% from [21], and 7.27% from [9], but performs worse than [19]. This work reduces the error rate of 2% in non-occlusion and 1% in occlusion conditions.

Table 2. Comparison of the error rate.

Work	KITTI2015	
	Noc (Error Rate)	Occ (Error Rate)
JSSC 2019 [20]	-	7.52%
TCSVT 2019 [21]	-	6.88%
TCSVT 2021 [18]	6.54%	7.44%
TCSVT 2021 [19]	5.66%	5.84%
TCAS-I 2021 [9]	6.58%	7.27%
This work	4.61%	6.04%

Meanwhile, this work aims at stereo vision post-processing with high-precision infrared speckle structured light in future work. The infrared camera ignores dark backgrounds and weakly textured backgrounds, providing a more accurate match to the original image. The real-world infrared speckles stereo image and obtained disparity map are shown in Figure 16. Figure 17 shows that the five-direction occlusion filling method proposed in this work has improved the edges of the image by FPGA implementation with MT9V034 global-shutter CMOS Image Sensors.

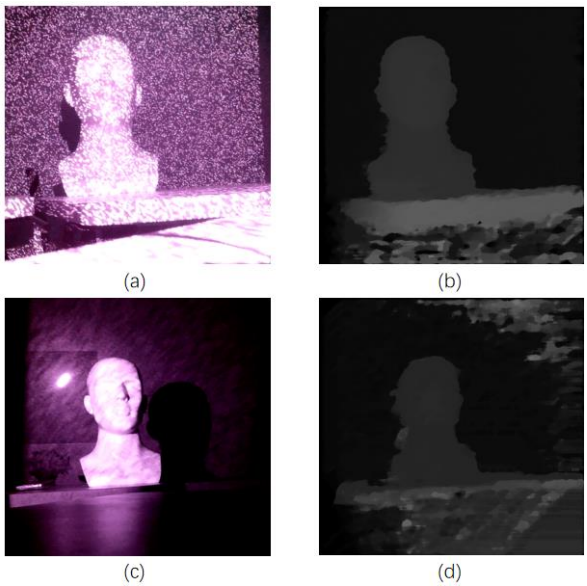

Figure 16. (**a**–**c**) image of real world obtained by camera; (**b**) Real time output disparity map of (**a**); (**d**) Real time output disparity map of (**c**).

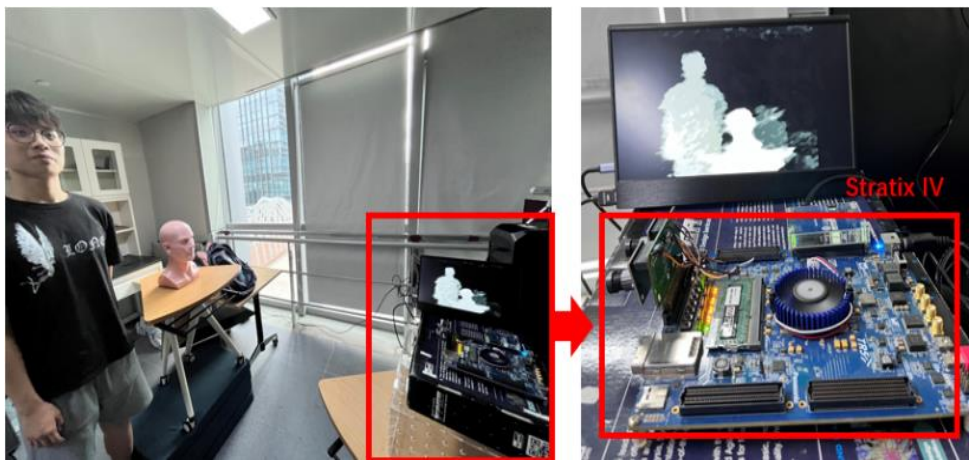

Figure 17. FPGA demonstration.

The hardware architecture based on our optimization algorithms is completed and implemented on the Stratix-IV platform, and it consumes about 5.6 K LUTs, 12.8 K registers, and 2.5 M bits of on-chip memory. The maximum working frequency can reach up to 98.28 MHz for the 640 × 480 resolution and 128 disparity range with a power dissipation of 1.459 W and a processing speed of 320 frames per Second. Compared with work [12] and [13], although our post-processing module consumes more than 4000 registers, it only uses less than 1/10 LUTs of their work and presents high disparity accuracy. Meanwhile, the memory usage in our work has been reduced to less than about 1/9 in [13] and even more than [12]. In addition, Stereo MATLAB Calibration Toolbox is used in the proposed hardware solution, meaning that it is a very general solution to be adopted after calculating the calibration parameters with the software tool most widely used for camera calibration.

5. Conclusions

In conclusion, this paper proposes a high-accuracy hardware-friendly architecture for post-processing a typical stereo vision Semi-Global Matching algorithm to improve disparity precession on FPGA platforms through subpixel interpolation and five-direction occlusion filling, after which the depth information is obtained as well through a floating-point operator. The proposal aims to enhance the accuracy of the disparity map for real cases.

The overall purpose of this disparity refinement processing is concluded as the following. (1) Initially, this paper proposes an interpolation algorithm obtaining a fractional disparity value with subpixel information instead of an integer pixel accuracy by adding a divided cosine look-up table and Newton's division operator. Accuracy is massively promoted by adding these essential fraction parts and copying the three decimal places after the decimal point according to the best precision. (2) Secondly, the left-right check module is utilized to correct the effects of both left and right disparity maps. (3) Then, to obtain an optimized disparity map with relatively low hardware resources, a five-direction occlusion filling considering pixel coordinates around the center is presented with different real situations. Based on that, according to the overlapping bubble ordering framework, we propose a parallel combinatorial logic that can efficiently get the arrangement permutations from small to large in those five directions simultaneously. (4) Consequently, the final disparity value is converted into more direct depth information for real sense by using a single precession floating operation, including floating-point multiplier and division.

This work has been verified on the FPGA platforms and compared with several advancing kinds of research, which are adequately hardware-friendly and convincing with the error rate of the output disparity map of nearly 4.61%, apparently superior to other work.

The deep neural network (DNN) often produces good depth estimation in the literature [22,23], while the hardware usage of an accelerator is incompatible with the requirement of real-time and low power. In the future, we may take advantage of the DNN method only for disparity refinement.

Author Contributions: Conceptualization, X.F.; methodology, Y.M.; software, Y.M., X.F., X.G. and K.L.; validation, Y.M., X.F. and X.G.; formal analysis, X.F. and X.G.; investigation, Y.M., X.G.; resources, X.F. and K.L.; data curation, Y.M., X.F., X.G. and K.L.; writing—original draft preparation, Y.M., X.F. and X.G.; writing—review and editing, Y.M.; visualization, Y.M. and X.F.; supervision, L.C. and F.A.; project administration, F.A.; funding acquisition, F.A. All authors have read and agreed to the published version of the manuscript.

Funding: This work was partially funded by the Science, Technology, and Innovation Commission of Shenzhen Municipality under grant JSGG20200102162401765, K2021390006, and K2021390007.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Lu, K.; Wang, X.; Wang, Z.; Wang, L. Binocular stereo vision based on OpenCV. In Proceedings of the IET International Conference on Smart and Sustainable City, Shanghai, China, 6–8 July 2011; pp. 1–4. [CrossRef]
2. Brown, M.Z.; Burschka, D.; Hager, G.D. Advances in computational stereo. *IEEE Trans. Pattern Anal. Mach. Intell.* **2003**, *25*, 993–1008. [CrossRef]
3. Shi, D.; Li, Y. Depth extraction method based on binocular stereo matching. In Proceedings of the 4th International Congress on Image and Signal Processing, Shanghai, China, 15–17 October 2011; pp. 1420–1423. [CrossRef]
4. Takaya, K. Stereo disparity measurement for binocular stereo video systems. In Proceedings of the ICCAS-SICE, Fukuoka, Japan, 18–21 August 2009; pp. 2648–2651.
5. Seki, A.; Okutomi, M. Robust Obstacle Detection in General Road Environment Based on Road Extraction and Pose Estimation. In Proceedings of the 2006 IEEE Intelligent Vehicles Symposium, Meguro-Ku, Japan, 13–15 June 2006; pp. 437–444. [CrossRef]
6. Zhao, Y.; Hou, X.; Jia, L.; Ma, S. The obstacle avoidance system for mobile robot based on binocular stereo vision. In Proceedings of the 8th World Congress on Intelligent Control and Automation, Jinan, China, 7–9 July 2010; pp. 6461–6465. [CrossRef]
7. Minaee, S.; Liang, X.; Yan, S. Modern Augmented Reality: Applications, Trends, and Future Directions. *arXiv* **2022**, arXiv:2202.09450. [CrossRef]
8. Hirschmuller, H. Accurate and efficient stereo processing by semi-global matching and mutual information. In Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), San Diego, CA, USA, 20–25 June 2005; Volume 2, pp. 807–814. [CrossRef]
9. Dong, P.; Chen, Z.; Li, Z.; Fu, Y.; Chen, L.; An, F. A 4.29nJ/pixel Stereo Depth Coprocessor with Pixel Level Pipeline and Region Optimized Semi-Global Matching for IoT Application. *IEEE Trans. Circuits Syst. I Regul. Pap.* **2021**, *69*, 334–346. [CrossRef]
10. Fan, R.; Ai, X.; Dahnoun, N. Road Surface 3D Reconstruction Based on Dense Subpixel Disparity Map Estimation. *IEEE Trans. Image Process.* **2018**, *27*, 3025–3035. [CrossRef] [PubMed]
11. Chen, Z.; Dong, P.; Li, Z.; Yao, R.; Ma, Y.; Fang, X.; Deng, H.; Zhang, W.; Chen, L.; An, F. Real-Time FPGA-Based Binocular Stereo Vision System with Semi-Global Matching Algorithm. In Proceedings of the 2021 IEEE 34th International System-on-Chip Conference (SOCC), Las Vegas, NV, USA, 14–17 September 2021; pp. 158–163. [CrossRef]
12. Jin, S.; Cho, J.; Dai Pham, X.; Lee, K.M.; Park, S.K.; Kim, M.; Jeon, J.W. FPGA Design and Implementation of a Real-Time Stereo Vision System. *IEEE Trans. Circuits Syst. Video Technol.* **2010**, *20*, 15–26. [CrossRef]
13. Cambuim, L.F.S.; Oliveira, L.A.; Barros, E.N.S.; Ferreira, A. An FPGA-based real-time occlusion robust stereo vision system using semi-global matching. *J. Real-Time Image Proc.* **2020**, *17*, 1447–1468. [CrossRef]
14. Zabih, R.; Woodfill, J. Non-parametric local transforms for computing visual correspondence. In *Lecture Notes in Computer Science*; Computer Vision—ECCV'94. ECCV 1994; Eklundh, J.O., Ed.; Springer: Berlin/Heidelberg, Germany, 1994; Volume 801.
15. Chen, B.; Chen, H.-P. A realization of mutual information calculation on GPU for semi-global stereo matching. In Proceedings of the 2012 Fifth International Conference on Intelligent Networks and Intelligent Systems, Tianjin, China, 1–3 November 2012; pp. 113–116. [CrossRef]
16. Haller, I.; Nedevschi, S. Design of Interpolation Functions for Subpixel-Accuracy Stereo-Vision Systems. *IEEE Trans. Image Process.* **2012**, *21*, 889–898. [CrossRef] [PubMed]
17. Hirschmuller, H. Stereo Processing by Semiglobal Matching and Mutual Information. *IEEE Trans. Pattern Anal. Mach. Intell.* **2008**, *30*, 328–341. [CrossRef] [PubMed]
18. Lu, Z.; Wang, J.; Li, Z.; Chen, S.; Wu, F. A resource-efficient pipelined architecture for real-time semi-global stereo matching. *IEEE Trans. Circuits Syst. Video Technol.* **2021**, *32*, 660–673. [CrossRef]
19. Lee, Y.; Kim, H. A High-Throughput Depth Estimation Processor for Accurate Semiglobal Stereo Matching Using Pipelined Inter-Pixel Aggregation. *IEEE Trans. Circuits Syst. Video Technol.* **2022**, *32*, 411–422. [CrossRef]
20. Li, Z.; Wang, J.; Sylvester, D.; Blaauw, D.; Kim, H.S. A 1920 × 1080 25-frames/s 2.4-TOPS/W low-power 6-D vision processor for unified optical flow and stereo depth with semi-global matching. *IEEE J. Solid-State Circuits* **2019**, *54*, 1048–1058. [CrossRef]
21. Zhang, X.; Sun, H.; Chen, S.; Song, L.; Zheng, N. NIPM-sWMF: Toward efficient FPGA design for high-definition large-disparity stereo matching. *IEEE Trans. Circuits Syst. Video Technol.* **2019**, *29*, 1530–1543. [CrossRef]
22. Chabra, R.; Straub, J.; Sweeney, C.; Newcombe, R.; Fuchs, H. StereoDRNet: Dilated Residual StereoNet. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 11778–11787. [CrossRef]
23. Xu, H.; Zhang, J. AANet: Adaptive Aggregation Network for Efficient Stereo Matching. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 1956–1965. [CrossRef]